AMENDMENTS TO THE CLAIMS

1. (Currently amended): A method for modifying computer program instructions during execution of those instructions, the method comprising computer-implemented steps of:

writing a first value into a memory location, ~~wherein~~ the first value representing ~~represents~~ a first instruction~~, and wherein the first instruction is a patch class~~ that is a particular type of unconditional instruction;

fetching the first instruction from the memory location;

executing the first instruction; and

while executing the first instruction, overwriting the first value by writing a second value into the memory location, ~~wherein~~ the second value representing ~~represents~~ a second instruction~~, and wherein the second instruction is a patch class~~ that is said particular type of unconditional instruction;

wherein the overwriting is [[may be]] concurrent with the execution of the first instruction[.]; and

the memory location being overwritten, while the first instruction is being executed, without producing unexpected results.

2. (Currently amended): The method according to claim 1, further comprising:

executing the first instruction by a first thread in a simultaneous multiprocessing (SMP) system that executes multiple different threads concurrently utilizing a plurality of processors; and

modifying the first instruction by a second thread in said SMP system concurrently with the first thread executing the first instruction.

~~refetching the first instruction from the memory location; and~~

~~reexecuting the first instruction~~

3. (Original): The method according to claim 1, further comprising:

fetching the second instruction from the memory location; and

executing the second instruction.

4. (Original): The method according to claim 1, further comprising:

reconciling a processor's execution pipeline with the memory location, wherein the reconciliation ensures that the second instruction will be fetched and executed from the memory location if the program subsequently returns to that memory location.

5. (Currently amended): The method according to claim 1, further comprising ~~wherein~~ the particular type of unconditional instruction being an unconditional branch type of instruction. ~~patch class instructions include no-operation instructions and branch instructions.~~

6. (Currently amended): The method according to claim 1, further comprising the particular type of instruction being only either a no-operation type of instruction or an unconditional branch type of instruction. ~~wherein the steps are implemented in a multiprocessor computer system.~~

7. (Currently amended): The method according to claim 1, further comprising the particular type of instruction being a no-operation type of instruction. ~~wherein the steps are implemented in a uniprocessor computer system.~~

8. (Currently amended): A computer program product in a computer readable medium for use in a data processing system, for modifying computer program instructions during execution of those instructions, the computer program product comprising:

instructions for writing a first value into a memory location, ~~wherein~~ the first value representing ~~represents~~ a first instruction~~, and wherein the first instruction is a patch class~~ that is a particular type of unconditional instruction; and

instructions for overwriting, while executing the first instruction, the first value by writing a second value into the memory location, ~~wherein~~ the second value representing ~~represents~~ a second instruction~~, and wherein the second instruction is a patch class~~ that is said particular type of unconditional instruction;

wherein the overwriting is [[may be]] concurrent with the execution of the first instruction[[.]]; and

the memory location being overwritten, while the first instruction is being executed, without producing unexpected results.

9. (Original): The computer program product according to claim 8, further comprising:

instructions for reconciling a processor's execution pipeline with the memory location, wherein the reconciliation ensures that the second instruction will be fetched and executed from the memory location if the program subsequently returns to that memory location.

10. (Currently amended): The computer program product according to claim 8, further comprising ~~wherein~~ the particular type of unconditional instruction being only either a no-operation type of instruction or an unconditional branch type of instruction. ~~patch class instructions include no-operation instructions and branch instructions.~~

11. (Currently amended): The computer program product according to claim 8, further comprising:

~~wherein the steps are implemented by a multithreaded program.~~

instructions for executing the first instruction by a first thread in a simultaneous multiprocessing (SMP) system that executes multiple different threads concurrently utilizing a plurality of processors; and

instructions for modifying the first instruction by a second thread in said SMP system concurrently with the first thread executing the first instruction.

12. (Currently amended): The computer program product according to claim 8, further comprising the particular type of unconditional instruction being an unconditional branch instruction. ~~wherein the steps are implemented by a single-threaded program.~~

13. (Currently amended): A system for modifying computer program instructions during execution of those instructions, the system comprising:

a writing component which writes a first value into a memory location, ~~wherein~~ the first value representing ~~represents~~ a first instruction~~, and wherein the first instruction is a patch class~~ that is a particular type of unconditional instruction;

a fetching component which fetches the first instruction from the memory location;

a processing component which executes the first instruction; and

an overwriting component which, while the first instruction is being executed by the processing component, overwrites the first value by writing a second value into the memory location, ~~wherein~~ the second value representing ~~represents~~ a second instruction~~, and wherein the second instruction is a patch class~~ that is said particular type of unconditional instruction;

wherein the overwriting is [[may be]] concurrent with the execution of the first instruction[[.]]; and

the memory location being overwritten, while the first instruction is being executed, without producing unexpected results.

14. (Currently amended): The system according to claim 13, further comprising:

a first thread in a simultaneous multiprocessing system (SMP) executing the first instruction, the (SMP) system executing multiple different threads concurrently utilizing a plurality of processors; and

a second thread in said SMP system modifying the first instruction concurrently with the first thread executing the first instruction.

~~a refetching component which refetches the first instruction from the memory location; and~~

~~a processing component which reexecutes the first instruction.~~

15. (Original): The system according to claim 13, further comprising:

a fetching component which fetches the second instruction from the memory location; and

a processing component which executes the second instruction.

16. (Original): The system according to claim 13, further comprising:

a reconciliation component which reconciles a processor's execution pipeline with the memory location, wherein the reconciliation ensures that the second instruction will be fetched and executed from the memory location if the program subsequently returns to that memory location.

17. (Currently amended): The system according to claim 13, further comprising the particular type of unconditional instruction being an unconditional branch type of instruction. ~~wherein the patch class instructions include no-operation instructions and branch instructions.~~

18. (Currently amended): The system according to claim 13, further comprising the particular type of unconditional instruction being a no-operation type of instruction. ~~wherein the components are part of a multiprocessor computer system.~~

19. (Currently amended): The system according to claim 13, further comprising the particular type of unconditional instruction only either a no-operation type of instruction or an unconditional branch type of instruction. ~~wherein the components are part of a uniprocessor computer system.~~